Ruby - Bug #1813

Threading seg fault (1.9.1-p129 Linux/Mac)

07/24/2009 05:55 AM - charlton (Charlton Wang)

Status:	Closed		
Priority:	Normal		
Assignee:	kosaki (Motohiro KOSAKI)		
Target version:	2.0.0		
ruby -v:	ruby 1.9.1p129 (2009-05-12 revision 23412) [i686-linux]	**Backport:**	

Description

=begin
I'm not sure if there is a thread stack limitation change that is new to 1.9.1 but the following code now causes a segfault on Linux and Illegal Instruction on the Mac. I know this code is somewhat ridiculous (call stack is attempting to go 1001 levels deep) but this seems to work fine in 1.8.6. If the code is executed outside of a thread, everything works fine as well.

```
t = Thread.new do
n = 1000
n.times do |i|
Object.class_eval <<EOF
define_method "foo#{i}" do
if i < n-1
puts i
self.send("foo#{i+1}")
else
puts "done"
end
end
EOF
end
foo0
end
t.join
```

Thanks,
Charlton
=end

Related issues:

Related to Ruby - Bug #2558: r24591 causes Segfault	**Closed**	**01/05/2010**
Related to Ruby - Bug #4983: Fiber□□□□□□□□□□□□□□□□□□	**Closed**	**07/06/2011**
Related to Ruby - Bug #3781: FIBER_USE_NATIVE □□□□□□□□□□□□□□□□□□	**Closed**	**09/02/2010**
Has duplicate Ruby - Bug #3286: segfault in method_missing -> method -> metho...	**Closed**	**05/13/2010**

Associated revisions

Revision 5a73c71d - 07/02/2011 07:59 PM - kosaki (Motohiro KOSAKI)

- thread_pthread.c (get_stack): add to a care of gurad page on Mac OS X. [Bug #1813] [ruby-core:24540]
- signal.c (ruby_signal): SIGBUS use alternative stack too.
- signal.c (sigbus): On Mac, thread stack overflow makes SIGBUS instead of SIGSEGV. thus, added stackoverflow check.
- signal.c (default_handler): get rid of compilation warning.
- signal.c (Init_signal): ditto.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@32369 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 70dd402c - 07/02/2011 09:18 PM - kosaki (Motohiro KOSAKI)

- thread_pthread.c (get_stack): pthread_attr_getstack() doesn't return stack start addres, but stack base address. Thus,

we need to add stack size for getting stack start address.
And, we don't have to decrease guard size twice.

- thread_pthread.c (thread_start_func_1): don't use inaccurate stack start guess if native_thread_init_stack() can be used.
[Bug #1813] [ruby-core:24540]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@32371 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision e4c452da - 07/10/2011 08:39 PM - Yutaka Kanemoto

- thread_pthread.c (get_stack): need to adjust stack addr for
[Bug #1813] on AIX.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@32511 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 25af5ee5 - 07/11/2011 02:49 PM - Yutaka Kanemoto

thread_pthread.c (get_stack): need to adjust stack addr for
[Bug #1813] on AIX. backported r32511 from trunk.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/branches/ruby_1_9_3@32519 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 07/24/2009 06:01 AM - charlton (Charlton Wang)

=begin
I should have realized that the 1.9 series uses native threads and I suspect that's likely the cause (presumably a stack size limitation?).
=end

#2 - 10/16/2009 10:19 AM - naruse (Yui NARUSE)

- Category set to core

- Status changed from Open to Assigned

- Assignee set to ko1 (Koichi Sasada)

=begin

=end

#3 - 01/27/2010 09:56 AM - rogerdpack (Roger Pack)

=begin
seems to work fine for me with 1.9.1/1.9.2
Is it still a problem?
=end

#4 - 01/27/2010 04:14 PM - naruse (Yui NARUSE)

- Status changed from Assigned to Feedback

- Assignee deleted (ko1 (Koichi Sasada))

=begin

=end

#5 - 01/27/2010 04:43 PM - naruse (Yui NARUSE)

- Category set to core

- Status changed from Feedback to Open

=begin
reproduced on both Ubuntu 8.04 x86_64 and FreeBSD 8.0 x86_64.
=end

#6 - 01/27/2010 04:59 PM - nobu (Nobuyoshi Nakada)

=begin
Seems like a stack overflow.
=end

#7 - 05/14/2010 01:27 AM - mame (Yusuke Endoh)

- Status changed from Open to Closed

- % Done changed from 0 to 100

=begin
This issue was solved with changeset r27789.
Charlton, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

=end

#8 - 09/02/2010 03:27 AM - ivan (Ivan Pirlik)

=begin
I am still getting "Illegal Instruction" in 1.9.2-p0 on OSX 10.5.7 (on Ubuntu 9.10 x86_64 I get a SystemStackError, as expected).
=end

#9 - 01/12/2011 08:24 AM - ibc (Iñaki Baz Castillo)

=begin
The given code produces a segmentation fault under ruby 1.9.2-p0 and 1.9.2-p136 (tested in Ubuntu 64 bits). In 1.8 it doesn't crash:

(eval):3: [BUG] Segmentation fault
ruby 1.9.2p136 (2010-12-25 revision 30365) [x86_64-linux]

-- control frame ----------
c:0917 p:---- s:1837 b:1837 l:001836 d:001836 CFUNC :puts
c:0916 p:---- s:1835 b:1835 l:001834 d:001834 CFUNC :puts
c:0915 p:0028 s:1831 b:1831 l:000978 d:001830 LAMBDA (eval):3
c:0914 p:---- s:1829 b:1829 l:001828 d:001828 FINISH
c:0913 p:0040 s:1827 b:1827 l:000978 d:001826 LAMBDA (eval):4
c:0912 p:---- s:1825 b:1825 l:001824 d:001824 FINISH
c:0911 p:0040 s:1823 b:1823 l:000978 d:001822 LAMBDA (eval):4
c:0910 p:---- s:1821 b:1821 l:001820 d:001820 FINISH
c:0909 p:0040 s:1819 b:1819 l:000978 d:001818 LAMBDA (eval):4
c:0908 p:---- s:1817 b:1817 l:001816 d:001816 FINISH
c:0907 p:0040 s:1815 b:1815 l:000978 d:001814 LAMBDA (eval):4
c:0906 p:---- s:1813 b:1813 l:001812 d:001812 FINISH
c:0905 p:0040 s:1811 b:1811 l:000978 d:001810 LAMBDA (eval):4
c:0904 p:---- s:1809 b:1809 l:001808 d:001808 FINISH
c:0903 p:0040 s:1807 b:1807 l:000978 d:001806 LAMBDA (eval):4
c:0902 p:---- s:1805 b:1805 l:001804 d:001804 FINISH
c:0901 p:0040 s:1803 b:1803 l:000978 d:001802 LAMBDA (eval):4

=end

#10 - 02/15/2011 10:58 PM - kosaki (Motohiro KOSAKI)

- Status changed from Closed to Open

- Target version set to 2.0.0

=begin
This issue can be still reproduced on trunk. (ruby 1.9.3dev (2011-02-15 trunk 30882) [x86_64-linux])

reopened.
=end

#11 - 06/26/2011 02:06 PM - naruse (Yui NARUSE)

- Status changed from Open to Assigned

- Assignee set to kosaki (Motohiro KOSAKI)

- % Done changed from 100 to 0

#12 - 07/03/2011 04:59 AM - kosaki (Motohiro KOSAKI)

- Status changed from Assigned to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r32369.

Charlton, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

- thread_pthread.c (get_stack): add to a care of gurad page on Mac OS X. [Bug #1813] [ruby-core:24540]
- signal.c (ruby_signal): SIGBUS use alternative stack too.
- signal.c (sigbus): On Mac, thread stack overflow makes SIGBUS instead of SIGSEGV. thus, added stackoverflow check.
- signal.c (default_handler): get rid of compilation warning.
- signal.c (Init_signal): ditto.

#13 - 07/03/2011 06:24 AM - kosaki (Motohiro KOSAKI)

Linux and MacOS X had completely different bugs. Thus I've commited two patches, r32369 and 32371.
Now, both platform raise SystemStackError correctly.

thanks.